

# Preface

Cryptography is concerned with communication and computation in the presence of adversaries. A fundamental challenge in theoretical and practical cryptography is to minimize the computational complexity of honest parties while providing security against computationally strong attackers. Ideally, one would like to construct cryptographic tools or “primitives” which can be computed extremely fast and retain strong security guarantees. These two targets, *efficiency* and *security*, are somewhat contradictory as highly efficient functions may be too simple to generate cryptographic hardness. Identifying the minimal level of efficiency which still guarantees security is therefore a major research goal.

This book studies this question through the lens of parallel-time complexity. We ask whether basic cryptographic primitives can be computed in constant parallel time. Formally, we consider the possibility of computing instances of these primitives using  $\text{NC}^0$  circuits, in which each output bit depends on a constant number of input bits. Despite previous efforts in this direction, there has been no convincing theoretical evidence supporting this possibility, which was posed as an open question in several previous works (e.g., [50, 69, 85, 105, 112]). We essentially settle this question by providing strong evidence for the possibility of cryptography in  $\text{NC}^0$ . In particular, we derive the following results.

**Existence of Cryptographic Primitives in  $\text{NC}^0$**  We show that many cryptographic primitives can be realized in  $\text{NC}^0$  under standard intractability assumptions used in cryptography, such as those related to factoring, discrete logarithm, or lattice problems. This includes one-way functions, pseudorandom generators, symmetric and public-key encryption schemes, digital signatures, message authentication schemes, commitment schemes, collision-resistant hash functions and zero-knowledge proofs. Moreover, we provide a *compiler* that transforms an implementation of a cryptographic primitive in a relatively “high” complexity class into an  $\text{NC}^0$  implementation. This compiler is also used to derive new unconditional  $\text{NC}^0$  *reductions* between different cryptographic primitives. In some cases, no parallel reductions of this type were previously known, even in  $\text{NC}$ . Interestingly, we get *non-black-box* reductions.

**Pseudorandom Generators with Linear Stretch in  $\text{NC}^0$**  The aforementioned constructions of pseudorandom generators (PRGs) were limited to stretching a seed of  $n$  bits to  $n + o(n)$  bits. This leaves open the existence of a PRG with a linear (let alone superlinear) stretch in  $\text{NC}^0$ . We construct a linear-stretch PRG in  $\text{NC}^0$  under a relatively new intractability assumption presented by Alekhnovich [5]. We also identify a new connection between such pseudorandom generators and hardness of approximations for combinatorial optimization problems. In particular, we show that an  $\text{NC}^0$  pseudorandom generator with linear stretch implies that Max 3SAT cannot be efficiently approximated to within some multiplicative constant. Our argument is quite simple and does not rely on PCP machinery.

**Cryptography with Constant Input Locality** After studying  $\text{NC}^0$  functions, in which each output bit depends on a constant number of input bits, we move on to study functions in which each *input* bit affects a constant number of output bits, i.e., functions with constant *input* locality. We characterize what cryptographic tasks can be performed with constant input locality. On the negative side, we show that primitives that require some form of non-malleability (such as digital signatures, message authentication, or non-malleable encryption) *cannot* be realized with constant input locality. On the positive side, assuming the intractability of certain problems from the domain of error correcting codes, we obtain new constructions of one-way functions, pseudorandom generators, commitments, and semantically secure public-key encryption schemes whose input locality is constant. Moreover, these constructions also enjoy constant *output* locality. Therefore, they give rise to cryptographic hardware that has constant-depth, constant fan-in and constant *fan-out*.

**A Study of Randomizing Polynomials** Most of our results make use of the machinery of *randomizing polynomials*, which were introduced by Ishai and Kushilevitz [92] in the context of information-theoretic secure multiparty computation. Randomizing polynomials allow us to represent a function  $f(x)$  by a low-degree randomized mapping  $\hat{f}(x, r)$  whose output distribution on an input  $x$  is a *randomized encoding* of  $f(x)$ . We present several variants of this notion along with new constructions. Our new variants have applications not only in the domain of parallel cryptography. For example, by extending the notion of randomizing polynomials to the computational setting, we show that, assuming a PRG in  $\text{NC}^1$ , the task of computing an *arbitrary* (polynomial-time computable) function with computational security can be reduced to the task of securely computing degree-3 polynomials (say, over  $\mathbb{F}_2$ ) without further interaction. This gives rise to new, conceptually simpler, constant-round protocols for general functions.

**This Version** This book is based on the author's doctoral dissertation which was submitted to the Technion in 2007. Some of the sections and proofs have been extended to provide more details and intuition. The content has also been updated to reflect the main recent developments in the field of parallel-time cryptography. A detailed chapter-by-chapter description of the contents and a high-level list of updates appear in Sects. 1.2.2 and 1.3.



<http://www.springer.com/978-3-642-17366-0>

Cryptography in Constant Parallel Time

Applebaum, B.

2014, XVI, 193 p. 3 illus., Hardcover

ISBN: 978-3-642-17366-0